Amendments to the Claims

This listing of claims will replace all prior versions and listings of claims in the application.

Listing of Claims:

1.-38. (Canceled)

39. (New) One or more processor-accessible storage media comprising processor-executable instructions that, when executed, direct a device to perform file compilation actions, comprising:

accepting a plurality of files, each file of the plurality of files corresponding to a respective file type and including source code, wherein at least two files have different file types;

instantiating a plurality of build providers, wherein each of the plurality of build providers is exclusively associated with one file type;

instantiating a single instance of a build provider host, wherein the build provider host includes a plurality of interfaces, wherein each of the plurality of interfaces is associated with one of the plurality of build providers during file compilation actions;

calling a file path interface of each of the plurality of build providers, wherein a path to each associated file is received at the respective build providers from the build provider host;

invoking a usable code language interface of each of the plurality of build providers, wherein each of the plurality of build providers provides a code language to be used during file compilation actions;

sequentially calling a generate code interface of each build provider, wherein each build provider contributes at least a portion of the source code of the build provider's associated file to be compiled via one or more of:

writing the source code to a code file object by calling a create code file object interface of the build provider host;

writing the source code to a stipulated path by calling a get code file path interface of the build provider host; and

generating a code compile unit by calling a get code object model provider interface and an add code compile unit interface of the build provider host;

ascertaining one or more resources of each file of the plurality of files via the associated build provider;

accessing a configuration file including a data structure that exclusively maps respective file types of the plurality of files to a respective build provider, wherein a new build provider is registered by updating the data structure of the configuration file to include a new entry that maps a new file type to the new build provider;

launching a compiler to compile the source code and the one or more resources of each file of the plurality of files into a single target assembly; and

using a build provider manager to control the plurality of build providers and the build provider host during file compilation actions.

40. (New) The one or more processor-accessible storage media as recited in claim 39, wherein the plurality of interfaces of the build provider host further includes one or more of: a get referenced assemblies interface; an add assembly reference interface; and a create embedded resource interface.

41. (New) The one or more processor-accessible storage media as recited in claim 40, wherein, upon execution: the get referenced assemblies returns a collection of one or more assemblies to be compiled; the add assembly reference adds at least one assembly to be referenced during compilation; the create code file object creates a file object that is to include a new source code for compilation; the get code file path returns a path to a file having source code to be included in compilation; the get code object model provider returns a code object model provider usable to generate a code compile unit; the add code compile unit enables one of the build providers to add another code compile unit to file compilation; and the create embedded resource creates a new resource to be added to file compilation.

42. (New) The one or more processor-accessible storage media as recited in claim 39, wherein calling the file path interface of each build provider further comprises one of: calling a physical file path; and calling a virtual file path.

43. (New) The one or more processor-accessible storage media as recited in claim 39, wherein each file type is specified by one of: an extension; a file naming scheme for an operating system; a file attribute; and a file tag.

44. (New) The one or more processor-accessible storage media as recited in claim 39, further comprising: determining the code language for one of the plurality of files; returning the code language if the code language is determined; and returning a null if the code language is language agnostic.

45. (New) The one or more processor-accessible storage media as recited in claim 39, wherein the source code of a respective file is one or more of: contiguous; discontinuous; including one or more modules; intermixed with other non-code portions; and directly or indirectly derived from non-code portions.

46. (New) The one or more processor-accessible storage media as recited in claim 39, wherein generating a code compile unit by calling a get code object model provider interface and an add code compile unit interface of the build provider host further comprises generating the code compile unit as a language-independent structure

47. (New) The one or more processor-accessible storage media as recited in claim 39, wherein the action of compiling further comprises constructing at least one of: an object code file; an executable file; a dynamically linked library file; and an intermediate language file.

48. (New) The one or more processor-accessible storage media as recited in claim 39, wherein ascertaining one or more resources of each file of the plurality of files via the associated build provider further comprises calling a create embedded resource interface of the build provider host.

49. (New) The one or more processor-accessible storage media as recited in claim 39, wherein at least a portion of the processor-executable instructions comprise at least part of an operating system.

50. (New) The one or more processor-accessible storage media as recited in claim 39, wherein at least a portion of the processor-executable instructions comprise at least part of a program that is capable of establishing a runtime environment.